LAYER-WISE CHARACTERIZATION OF LATENT INFORMATION LEAKAGE IN FEDERATED LEARNING

Fan Mo, Anastasia Borovykh, Mohammad Malekzadeh, Hamed Haddadi & Soteris Demetriou
Imperial College London
{f.mo18,a.borovykh,m.malekzadeh,h.haddadi,s.demetriou}@imperial.ac.uk

ABSTRACT

Training deep neural networks via federated learning allows clients to share, instead of the original data, only the model trained on their data. Prior work has demonstrated that in practice a client's private information, unrelated to the main learning task, can be discovered from the model's gradients, which compromises the promised privacy protection. However, there is still no formal approach for quantifying the leakage of private information via the shared updated model or gradients. In this work, we analyze property inference attacks and define two metrics based on (i) an adaptation of the *empirical $\mathcal{V}$-information*, and (ii) a *sensitivity analysis* using Jacobian matrices allowing us to measure changes in the gradients with respect to latent information. We show the applicability of our proposed metrics in localizing private latent information in a layer-wise manner and in two settings where (i) we have or (ii) we do not have knowledge of the attackers' capabilities. We evaluate the proposed metrics for quantifying information leakage on three real-world datasets using three benchmark models.

1 INTRODUCTION

Federated learning (FL), allows *clients* to jointly train a model, *e.g.* a deep neural network (DNN), on their local data, and iteratively share their updates with a *server* that aggregates the received updates (McMahan et al., 2017). There is a surge of interest in FL (Kairouz et al., 2019; Bonawitz et al., 2017; Hard et al., 2018), as it resolves the need for collecting private data in a centralized location which in many cases is not possible due to computational costs, privacy risks, and even legal considerations (Kairouz et al., 2019; Bonawitz et al., 2019). However, it is found that even sharing the gradients of private data *w.r.t.* a DNN's parameters is not a privacy panacea as these gradients can contain private information (Melis et al., 2019; Zhu et al., 2019; Nasr et al., 2019).

The *property inference attacks* (PIA) (Melis et al., 2019; Wang et al., 2019) in FL aim to infer some private information about a targeted client, *e.g.* a client's property like gender or race. Although there are some works (Neyshabur et al., 2017; Lee et al., 2020; Arora et al., 2019; Achille et al., 2019) on the memorization and generalization of DNNs *w.r.t.* the main task that DNNs are trained on, these works are not able to explain the memorization of latent information independent of the main task. There also are some works on (layer-wise) understanding of the learned representation by a DNN (Zeiler & Fergus, 2014; Mahendran & Vedaldi, 2015; Shwartz-Ziv & Tishby, 2017; Saxe et al., 2019), but their proposed methods only provide insights on how information evolves during *forward propagation*, while the analysis of information captured in the *backward propagation* is still missing.

Contributions. We focus on the open problem of how PIA leverages latent information presented in the computed gradients, and show that the Shannon mutual information (Shannon, 1948) cannot properly quantify the risk of PIA. We propose adopting a more generalized notion of information, *$\mathcal{V}$-information* (Xu et al., 2020), also known as 'usable' information. This can measure the layer-wise latent information privacy risk and explain which layers are the most vulnerable to PIA. Furthermore, as $\mathcal{V}$-information requires knowledge of the specific family of attack models, we further present a metric based on a *sensitivity analysis* of the gradients *w.r.t.* latent information, which is more suitable when modeling a general adversary.

2 PRIVACY ANALYSIS: METRICS FOR QUANTIFYING PROPERTY PRIVACY

Threat Model. We aim at characterizing *property privacy* (Melis et al., 2019; Hitaj et al., 2017; Zhu et al., 2019), which is one type of latent information privacy, as opposed to *input privacy* (Bonawitz et al., 2017; Gu et al., 2018) that refers to the privacy of training data. Our focus is on the properties (*e.g.* gender or race) that can be inferred from input data (*e.g.* face images). As the property of a client in FL can even be inferred from the average gradients computed on the entire client's data, it is much more challenging to protect such latent information, compared to preventing pixel-wise reconstruction of the client's input data. We assume the PIA adversary $\mathcal{A}$ (*e.g.* the server or a malicious client) aims to disclose the property of a targeted client (*i.e.* victim) by observing gradients or updated models broadcast by the server (Melis et al., 2019). We assume the client's private properties are irrelevant to the main task of FL. To conduct the attack, we assume $\mathcal{A}$ can observe multiple model updates released by the victim and $\mathcal{A}$ has access to some (public) auxiliary data. Thus, $\mathcal{A}$ can train an *attack model*, *e.g.* a binary classifier (Melis et al., 2019), using the auxiliary data and the collected model updates, to discover the victim's property.

Shannon Information. A well-known metric for quantifying the information flow of the training dataset in a DNN's forward propagation is the Shannon mutual information (MI) (Shwartz-Ziv & Tishby, 2017; Goldfeld et al., 2019). Let X, Y, and $\hat{\mathrm{Y}}$ be the input, the ground-truth label, and the output of the DNN, respectively. Let T_l denote the intermediate representation in layer l (we refer Appendix A.1 for notations). The MI between X (or Y) and T_l satisfies the Data Processing Inequality (DPI) (Tishby et al., 2000),

$$I(\mathrm{X};\mathrm{T}_1) \geq I(\mathrm{X};\mathrm{T}_2) \cdots \geq I(\mathrm{X};\mathrm{T}_L) \geq I(\mathrm{X};\hat{\mathrm{Y}}),$$
$$I(\mathrm{Y};\mathrm{X}) \geq I(\mathrm{Y};\mathrm{T}_1) \geq I(\mathrm{Y};\mathrm{T}_2) \cdots \geq I(\mathrm{Y};\mathrm{T}_L) \geq I(\mathrm{Y};\hat{\mathrm{Y}}).$$

These inequalities correspond to the intuition that information should *not* increase during the layer-by-layer forward propagation in DNNs. However, DPI fails to explain the backward propagation because the Markov chain construction on inputs, model parameters, and computed *gradients* is much more complex than that in the forward propagation (see Figure 4 in Appendices). One can reasonably expect a DNN to extract features that are useful for the target task such that 'usable' information for the task Y memorized in DNN parameters/gradients, *i.e.* what is relevant for PIAs, would be *increasing* throughout the layers.

$\mathcal{V}$-Information. As claimed in Xu et al. (2020), Shannon MI, due to its assumption of unbounded computational power, fails to explain that the 'usable information' in the late layers of a DNN may be higher than in early layers. Such a definition of usable information can be highly relevant to understanding PIA. We, therefore, argue that the notion $\mathcal{V}$-information from Xu et al. (2020) is better suited for analyzing the latent information captured in the gradients of each DNN's layer. Here, we formulate PIA in the notion of $\mathcal{V}$-information.

Let the adversary, due to computational constraints, only has access to the attack models $f_{\mathcal{A}}$ from a specific predictive family $\mathcal{V}_{\mathcal{A}}$ (*e.g.* a family of random forest algorithm). Let $\mathrm{p} \in \{0,1\}$ be the (binary) property and $\mathrm{G}_l \in \mathbb{R}^{N_l \times N_{l-1}}$ be the layer l's gradients, taking values in the sample space, where $N_l \times N_{l-1}$ is the dimensionality of used gradients. Let g_l denote an instance of G_l and $f_{\mathcal{A}}[\mathrm{g}_l] \in [0,1]$ be a probability measure on the value of p computed via the side information g_l. Thus $f_{\mathcal{A}}[\mathrm{g}_l](\mathrm{p}_i) \in [0,1]$ denotes the probability of $\mathrm{p} = i$, for $i \in \{0,1\}$, that is achieved via the softmax of the $f_{\mathcal{A}}$'s output. Let $\mathcal{D}$ be a training dataset including samples of $\{\mathrm{g}_l, \mathrm{p}\}$. Let $\mathrm{g}_{l,i}$ be the layer l's gradients for data sample i. The *empirical $\mathcal{V}$-information* from G_l to p (*i.e.* information about p available in G_l) is defined as

$$\hat{I}_{\mathcal{V}_{\mathcal{A}}}(\mathrm{G}_l \to \mathrm{p}; \mathcal{D}) = \inf_{f_{\mathcal{A}} \in \mathcal{V}_{\mathcal{A}}} \frac{1}{|\mathcal{D}|} \sum_{\mathrm{p}_i \in \mathcal{D}} -\log f_{\mathcal{A}}[\emptyset](\mathrm{p}_i) - \inf_{f \in \mathcal{V}_{\mathcal{A}}} \frac{1}{|\mathcal{D}|} \sum_{\mathrm{g}_{l,i}, \mathrm{p}_i \in \mathcal{D}} -\log f_{\mathcal{A}}[\mathrm{g}_{l,i}](\mathrm{p}_i). \quad (1)$$

In Section 3, we show how to use the empirical $\mathcal{V}$-information to measure how well a property can be predicted when the adversary has access to the gradients of a layer as the side information.

Sensitivity. A downside of $\mathcal{V}$-information is that it relies on certain assumptions on the adversary's power; in particular the predictive family $\mathcal{V}_{\mathcal{A}}$. This makes $\mathcal{V}$-information practically similar

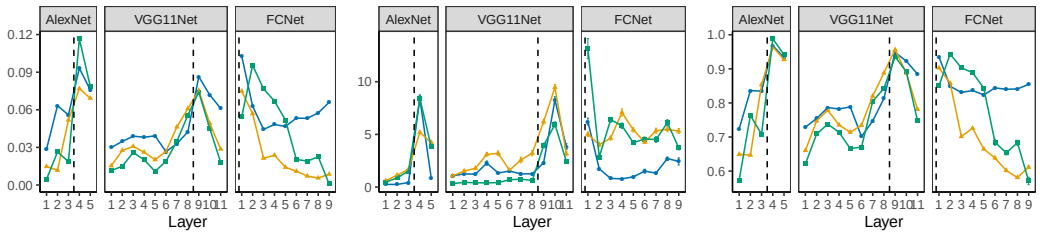

Figure 1: Results of privacy measurements *i.e.* (Left) $\mathcal{V}$-information, (Middle) sensitivity with F-norm, and (Right) AUC scores of PIA on each layer of AlexNet, VGG11Net, and FCNet, using LFW (—), CelebA (—), PubFig (—) dataset. Dashed lines (- -) refer to the connection from a Conv layer to an FC layer. Each trial runs 20 times, and error bars are 95% confidence interval.

to computing the adversary's area under curve (AUC). Therefore we also propose to directly compute privacy risk information for a *general* attacker, using solely the gradient information without making explicit assumptions about the attackers' model. We utilize the Jacobian matrix of the gradients as a straightforward *sensitivity* metric, similar to input-output Jacobian in Novak et al. (2018) and Sokolić et al. (2017), for measuring the general privacy risk on gradients *w.r.t.* high-level features (*e.g.* property). Intuitively, if certain gradients are non-sensitive to the input, or the property of data, the success of the attack can be expected to be lower. We compute the Jacobian of gradients *w.r.t.* the main task y and we expect the Jacobian of gradients *w.r.t.* a private property p to be similar to that on $\hat{\mathrm{y}}$, since both of them are high-level features.

We can use the Frobenius norm (F-norm) for matrices and the 1-norm or the ∞-norm for vectors in order to capture adversaries with different capabilities. Because Jacobians are compared across layers, the magnitude and size of the layer's parameters can also have an impact on the comparison results so we normalize the *norm of Jacobian* with G_l's size. Thus, given K data samples, we compute the 'Jacobian p-norm' averaged over the data samples as the privacy risk,

$$\frac{1}{K\psi_l}\sum_{k=1}^{K}\left\|\frac{\mathbf{J}_l^{(\mathrm{G})}(\hat{\mathrm{Y}}_k)}{\mathrm{range}(\mathrm{g}_l(\hat{\mathrm{Y}}_k))}\right\|_p \tag{2}$$

where $\mathbf{J}_l^{(\mathrm{G})}(\hat{\mathrm{Y}}) = \frac{\partial \mathrm{g}_l(\hat{\mathrm{Y}})}{\partial \hat{\mathrm{Y}}} = \frac{\partial}{\partial \hat{\mathrm{Y}}}\left(\frac{\partial \ell(\hat{\mathrm{Y}}, \boldsymbol{\theta}_l)}{\partial \boldsymbol{\theta}_l}\right)$ and $\mathrm{g}_l(.)$ represents the function that produces layer l's gradients $\mathrm{G}_l \in \mathbb{R}^{N_l \times N_{l-1}}$ with true output $\hat{\mathrm{Y}}$, $\ell(.)$ is the loss function over $\hat{\mathrm{Y}}$ and $\boldsymbol{\theta}$ (parameters of the complete model), so $\mathrm{g}_l(.)$ can be regarded as the partial derivative of $\ell(.)$ *w.r.t.* layer l's parameters $\boldsymbol{\theta}_l$ (*i.e.* backward propagation). Function range$(.)$ returns the range of values in one vector (*i.e.* $\max(.) - \min(.)$), and $p = F, 1$, or ∞. Parameter ψ_l is the normalization factor based on the size of the Jacobian matrix of layer l under each p. Thus, $\psi_l = \sqrt{N_l \times N_{l-1}},\ N_l \times N_{l-1}$, and 1 for $p = F,\ 1,$ and ∞, respectively.

3 Empirical Characterization and Validation

For our empirical evaluation we consider a scenario where we have a number of distributed devices, as e.g. in mobile settings or smart cities, and study the information leakage on several real-world models: variational Alexnet (Krizhevsky et al., 2012) and VGG11Net (Simonyan & Zisserman, 2014). We use Labeled Faces in the Wild (LFW) (Huang et al., 2008), Large-scale CelebFaces Attributes (CelebA) (Liu et al., 2015), and Public Figures Face Database (PubFig) (Kumar et al., 2009). Due to space limitations we refer an interested reader to Appendix A.4 for more details.

Layer-wise privacy characterization. The $\mathcal{V}$-information and sensitivity are measured on each layer of three models, AlexNet, VGG11Net, and FCNet, trained on three datasets, LFW, CelebA, and PubFig, respectively. For the three datasets, the main task of FL is to learn 'Gender', 'Glasses', and 'Gender', respectively, and the test accuracy using FedSGD reaches 99%, 99%, 94% for LFW, 87%, 82%, 84% for CelebA, and 97%, 98%, 92% for PubFig. As shown in Figure 1, the PIA's AUC scores have similar patterns with the prediction of both $\mathcal{V}$-information and sensitivity; *i.e.* the first FC layers have the highest private risk in terms of property information. More specifically, they both

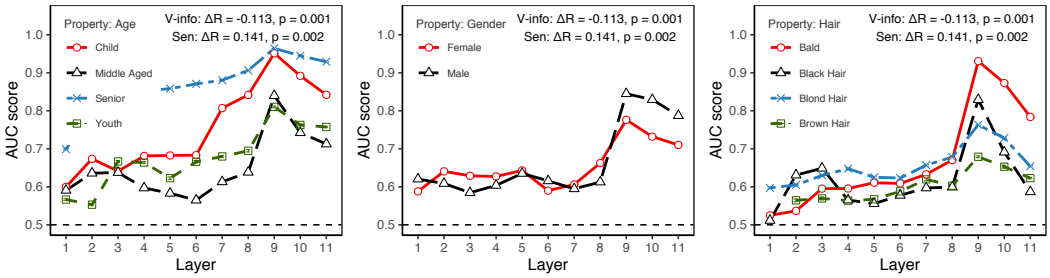

Figure 2: AUC scores of PIA aiming at (Left) Age, (Middle) Gender, and (Right) Hair on each layer of VGG11Net trained on LFW with Glasses as the main task. Pearson correlations between the AUC score and privacy metrics. Each trail runs 20 times .

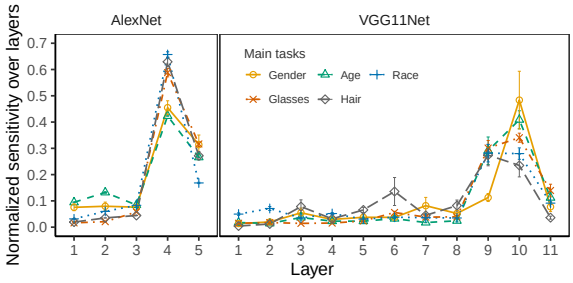

Figure 3: Normalized sensitivity (F-norm) over all layers of one model. Each trail runs 20 times.

have a similar pattern, showing that for these three models, the 4^{th}, 9^{th}/10^{th}, and 1^{st} layer have the highest privacy risk respectively.

Validation on multiple properties. Figure 2 illustrates the cases where the main task of FL is the classification of 'Glasses', and the PIA aims to infer properties of *Age*, *Gender*, and *Hair*. We observe similar patterns for PIAs on all properties. That is, the 9^{th} layer, *i.e.* the first FC layer after the last Conv layer in VGG11Net, still leaks the largest amount of property information. Furthermore, the correlation coefficient (R with significance level p) between $\mathcal{V}$-information/Sensitivity and attack AUC scores are given in the Figure, and the ΔR refers to coefficient changes when measuring other properties (*i.e.* AUC scores in Figure 1). We observe a decreasing prediction capability of $\mathcal{V}$-information on the other property, and an increased capability of sensitivity. Figure 3, trained on LFW, shows that no matter what the main classification task is, the first FC layers always have the highest (normalized) sensitivity. Moreover, we observe that, with the first FC layer of both networks (layer 4^{th} of AlexNet and layer 9^{th} of VGG11Net), 'Gender' has a lower sensitivity than the others, while in the next FC layer, situation changes; 'Gender' has a high sensitivity. As 'Gender' is a more abstract feature than 'Hair' color or whether 'Glasses' exist, the results indicate that a more high-level property tends to be more sensitive in latter layers compared to other properties.

4 Conclusion and Future Work

In this work, we proposed two mathematically-motivated metrics, $\mathcal{V}$-information and sensitivity, and showed that these allow to quantify latent information leakages in FL. We preliminarily validated their performance on layer-wise privacy characterization.

In future work we aim to explore i) applying the metrics on other types of datasets including time series or texts, ii) settings considering different levels of aggregations such as with more participating clients, iii) how the metrics could benefit the design of a defense mechanism leveraging layer-based privacy measurements (*e.g.* Mo & Haddadi (2019); McMahan et al. (2018)). In combination with the proposed metrics, we will provide a better understanding of *when* a model can leak sensitive information in FL and reveal opportunities to design flexible defenses for a better trade-off between privacy guarantees and cost.

REFERENCES

Alessandro Achille, Giovanni Paolini, and Stefano Soatto. Where is the information in a deep neural network? *arXiv preprint arXiv:1905.12213*, 2019.

Yoshinori Aono, Takuya Hayashi, Lihua Wang, Shiho Moriai, et al. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security*, 13(5):1333–1345, 2017.

Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. In *Advances in Neural Information Processing Systems*, pp. 8141–8150, 2019.

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a cpu and gpu math expression compiler. In *Proceedings of the Python for scientific computing conference (SciPy)*, volume 4, pp. 1–7. Austin, TX, 2010.

Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1175–1191. ACM, 2017.

Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečnỳ, Stefano Mazzocchi, H Brendan McMahan, et al. Towards federated learning at scale: System design. In *Twelve Conference on Machine Learning and Systems*, 2019.

Ziv Goldfeld, Ewout Van Den Berg, Kristjan Greenewald, Igor Melnyk, Nam Nguyen, Brian Kingsbury, and Yury Polyanskiy. Estimating information flow in deep neural networks. In *International Conference on Machine Learning*, pp. 2299–2308, 2019.

Zhongshu Gu, Heqing Huang, Jialong Zhang, Dong Su, Hani Jamjoom, Ankita Lamba, Dimitrios Pendarakis, and Ian Molloy. Yerbabuena: Securing deep learning inference data via enclave-based ternary model partitioning. *arXiv preprint arXiv:1807.00969*, 2018.

Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. Deep models under the gan: information leakage from collaborative deep learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 603–618. ACM, 2017.

Gary B Huang, Marwan Mattar, Tamara Berg, and Eric Learned-Miller. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. 2008.

Bargav Jayaraman and David Evans. Evaluating differentially private machine learning in practice. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pp. 1895–1912, 2019.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

Neeraj Kumar, Alexander C Berg, Peter N Belhumeur, and Shree K Nayar. Attribute and simile classifiers for face verification. In *2009 IEEE 12th international conference on computer vision*, pp. 365–372. IEEE, 2009.

Jaehoon Lee, Samuel S Schoenholz, Jeffrey Pennington, Ben Adlam, Lechao Xiao, Roman Novak, and Jascha Sohl-Dickstein. Finite versus infinite neural networks: an empirical study. *arXiv preprint arXiv:2007.15801*, 2020.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5188–5196, 2015.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pp. 1273–1282, 2017.

H Brendan McMahan, Galen Andrew, Ulfar Erlingsson, Steve Chien, Ilya Mironov, Nicolas Papernot, and Peter Kairouz. A general approach to adding differential privacy to iterative training procedures. *Privacy Preserving Machine Learning (CCS Workshop)*, 2018.

Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 691–706. IEEE, 2019.

Fan Mo and Hamed Haddadi. Efficient and private federated learning using tee. In *EuroSys*, 2019.

Fan Mo, Ali Shahin Shamsabadi, Kleomenis Katevas, Soteris Demetriou, Ilias Leontiadis, Andrea Cavallaro, and Hamed Haddadi. Darknetz: towards model privacy at the edge using trusted execution environments. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*, pp. 161–174, 2020.

Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 739–753. IEEE, 2019.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems*, pp. 5947–5956, 2017.

Roman Novak, Yasaman Bahri, Daniel A Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. Sensitivity and generalization in neural networks: an empirical study. In *International Conference on Learning Representations (ICLR)*, 2018.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

Nicolas Pinto, Zak Stone, Todd Zickler, and David Cox. Scaling up biologically-inspired computer vision: A case study in unconstrained face recognition on facebook. In *CVPR 2011 WORKSHOPS*, pp. 35–42. IEEE, 2011.

David Martin Powers. Evaluation: From precision, recall and f-measure to roc, informedness, markedness & correlation. *Journal of Machine Learning Technologies*, 2(1):37–63, 201.

Andrew M Saxe, Yamini Bansal, Joel Dapello, Madhu Advani, Artemy Kolchinsky, Brendan D Tracey, and David D Cox. On the information bottleneck theory of deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124020, 2019.

Claude E Shannon. A mathematical theory of communication. *Bell system technical journal*, 27(3): 379–423, 1948.

Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Jure Sokolić, Raja Giryes, Guillermo Sapiro, and Miguel RD Rodrigues. Robust large margin deep neural networks. *IEEE Transactions on Signal Processing*, 65(16):4265–4280, 2017.

Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. Beyond inferring class representatives: user-level privacy leakage from federated learning. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pp. 2512–2520. IEEE, 2019.

Wenqi Wei, Ling Liu, Margaret Loper, Ka-Ho Chow, Mehmet Emre Gursoy, Stacey Truex, and Yanzhao Wu. A framework for evaluating gradient leakage attacks in federated learning. *arXiv preprint arXiv:2004.10397*, 2020.

Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. A theory of usable information under computational constraints. In *International Conference on Learning Representations (ICLR)*, 2020.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pp. 268–282. IEEE, 2018.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pp. 818–833. Springer, 2014.

Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. In *Advances in Neural Information Processing Systems*, pp. 14747–14756, 2019.

A Appendix

A.1 Notations

Throughout this paper, we use lower-case italic, *e.g.* x, y, for deterministic scalar values; lower-case bold italic, *e.g.* $\boldsymbol{x}, \boldsymbol{y}$, for deterministic vectors; upper-case bold italic, *e.g.* $\boldsymbol{X}, \boldsymbol{Y}$, for deterministic matrices. We use lower-case normal, *e.g.* x, y, for instances of a random variable, and upper-case normal, *e.g.* X, Y, for random variables of any dimensions.

A.2 Gradient computation

Here we give more background on the gradient computations and specifically how the *backward* pass can reveal sensitive information.

We first revisit how gradients are computed as a function of the input to which later shows the root cause for how a gradient $\boldsymbol{G}$ can reveal input data $\boldsymbol{X}$ and information about the property p. Let $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_K]$ be one batch of data consisting of K samples from client c's training dataset $\mathcal{X}^c$, and let $\boldsymbol{Y} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_K]$ be the corresponding ground truth (*e.g.* labels for a classification task). The complete training dataset of all C clients is denoted by $\mathcal{X} = \{\mathcal{X}^1, ..., \mathcal{X}^C\}$. For a DNN model with L layers, we denote layer l's parameters with $\boldsymbol{W}_l$ and $\boldsymbol{b}_l$; the weights and biases, respectively. In the *forward propagation* from layer 1 to L, the layer l computes

$$\boldsymbol{A}_l = [\boldsymbol{a}_{l,1}, \ldots, \boldsymbol{a}_{l,K}] = \boldsymbol{W}_l \boldsymbol{T}_{l-1} + \boldsymbol{b}_l \boldsymbol{1}^\mathsf{T},$$

where $\boldsymbol{W}_l \in \mathbb{R}^{N_l \times N_{l-1}}$, $\boldsymbol{T}_{l-1} = [\boldsymbol{t}_{l-1,1}, \ldots, \boldsymbol{t}_{l-1,K}] \in \mathbb{R}^{N_{l-1} \times K}$, $\boldsymbol{b}_l \in \mathbb{R}^{N_l}$, $\boldsymbol{1} \in \{1\}^K$. N_l denotes the size (*i.e.* the number of neurons) of the layer l, and $\boldsymbol{T}_{l-1}$ shows the *intermediate representation* that is the output of the (previous) layer $l-1$, and $\boldsymbol{t}_{l-1,k}$ corresponds to one sample k's outputs. Then, the layer l's output is $\boldsymbol{T}_l = [\sigma(\boldsymbol{a}_{l,1}), \ldots, \sigma(\boldsymbol{a}_{l,K})] \in \mathbb{R}^{N_l \times K}$, denoted as $\sigma(\boldsymbol{A}_l)$ for simplicity, where $\sigma(\cdot)$ is the chosen activation function. Note that $\boldsymbol{T}_0 = \boldsymbol{X}$ and $\boldsymbol{T}_L = \hat{\boldsymbol{Y}}$ the prediction on ground truth.

In the *backward propagation*, the loss ℓ between $\hat{\boldsymbol{Y}}$ and $\boldsymbol{Y}$ propagates from the layer L to 1. For layer l, the gradient vector $\boldsymbol{G}_l$ consists of the gradients of the weights and biases $\{\boldsymbol{G}_l^{(w)}, \boldsymbol{G}_l^{(b)}\}$, which are computed using chain rule:

$$\boldsymbol{G}_l^{(w)} = \frac{\partial \ell}{\partial \boldsymbol{W}_l} = \frac{\partial \ell}{\partial \boldsymbol{A}_l} \frac{\partial \boldsymbol{A}_l}{\partial \boldsymbol{W}_l} = \frac{\partial \ell}{\partial \boldsymbol{A}_l} \boldsymbol{T}_{l-1}^\mathsf{T} \tag{3}$$

$$\boldsymbol{G}_l^{(b)} = \frac{\partial \ell}{\partial \boldsymbol{b}_l} = \frac{\partial \ell}{\partial \boldsymbol{A}_l} \frac{\partial \boldsymbol{A}_l}{\partial \boldsymbol{b}_l} = \frac{\partial \ell}{\partial \boldsymbol{A}_l} \mathbf{1} \tag{4}$$

where $\boldsymbol{G}_l^{(w)} \in \mathbb{R}^{N_l \times N_{l-1}}$ and $\boldsymbol{G}_l^{(b)} \in \mathbb{R}^{N_l}$ and have the same size with $\boldsymbol{W}_l$ and biases $\boldsymbol{b}_l$ respectively.

The set of all layers' $\boldsymbol{G}^{(w)}$ and $\boldsymbol{G}^{(b)}$ in Equation 3 and 4 is the minimum unit to update in FL, *i.e.* corresponding to the FedSGD. $\boldsymbol{G}^{(w)}$ and $\boldsymbol{G}^{(b)}$ are also targeted by adversaries to extract private information of $\boldsymbol{X}$ or its sub-information *e.g.* p. In FedAvg, before updating, more batches of data are fed for training, and the gradients of multiple batches are aggregated together element-wisely by

$$\sum_{\boldsymbol{X}_i \in \mathcal{X}^c} \{\boldsymbol{G}_l^{(w)}\}_{\boldsymbol{X}_i}; \sum_{\boldsymbol{X}_i \in \mathcal{X}^c} \{\boldsymbol{G}_l^{(b)}\}_{\boldsymbol{X}_i} \tag{5}$$

A.3 General analysis on gradients' privacy risks

We first directly analyze how $\boldsymbol{G}$ leaks information of $\boldsymbol{X}$ or p based on Equation 3, 4, and 5. Three *indications* for information leakage from DNN parameters (*i.e.* gradients) are: (1) One can obtain the layer l's input ($\boldsymbol{T}_{l-1}$) based on this layer's gradient updates. Because the first layer's input (*i.e.* $\boldsymbol{T}_0$) is $\boldsymbol{X}$, the original input is highly likely to be leaked, which has been shown in previous research (Aono et al., 2017). (2) Batch-based SGD (*i.e.* Minibatch) reduces the potential leakage on one specific data sample $\boldsymbol{x}_i, i \in 1, ..., K$, because gradients are computed over all samples in one batch (see Equation 3 and 4). Aggregation, *i.e.* the summation of gradients over multiple batches, also reduces the potential leakage (see Equation 5). (3) The term $\frac{\partial \ell}{\partial \boldsymbol{A}_l} = \frac{\partial \ell}{\partial \boldsymbol{T}_l} \odot \sigma'(\boldsymbol{a}_l) = \boldsymbol{W}_{l+1}^{\mathsf{T}} \frac{\partial \ell}{\partial \boldsymbol{A}_{l+1}} \odot \sigma'(\boldsymbol{A}_l)$ (note that $\odot$ is the Hadamard product), indicating that the gradient can potentially contain information of $\boldsymbol{A}_l$, $\boldsymbol{W}_{l+1}$, $\boldsymbol{b}_{l+1}$, and parameters of following layers (*i.e.* $\boldsymbol{A}_{l+1}, ..., \boldsymbol{A}_L$). However, the DPI (from layer 1 to L) indicates that T_{l+1} (or $\boldsymbol{a}_{l+1}$ here) cannot contain more information on X than T_l (or $\boldsymbol{a}_l$) in *forward propagation* (Saxe et al., 2019; Shwartz-Ziv & Tishby, 2017; Tishby et al., 2000).

Backward Markov chain. Similar to the Markov chain in the forward propagation, here we present the Markov chain in *backward propagation* from layer L to layer 1 based on how gradients are computed to clarify the information leakage from gradients. As shown in Figure 4, G_l is produced based on layer l's weights, biases, the previous representation, and next gradient (denoted by $\mathrm{W}_l, \mathrm{B}_l, \mathrm{T}_{l-1}$, and G_{l+1}). Previous research (Shwartz-Ziv & Tishby, 2017; Saxe et al., 2019) has indicated that, due to the data processing inequality, T_l contains more information about X then T_k with $k > l$. Furthermore, matrix $\boldsymbol{W}_l$ and $\boldsymbol{b}_l$ are linearly aggregated gradient updates from multiple previous $\boldsymbol{G}_l$ (*i.e.* the model parameters $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \boldsymbol{G}_t$ for the t^{th} batch data), so intuitively W_l and B_l possibly not contain more information on X than G_l, and then the information of X in T_{l-1} may reflect that of X in G_l.

Moreover, related to indication (2), when it comes to a particular x, W and B can be regarded as less sensitive if they have been updated based on the complete dataset $\mathcal{X}$. We further argue that the attempt on evaluating gradient leakage on original data $\mathrm{x}_i, i \in 1, ..., K$, such as (Wei et al., 2020), could be very limited and constrained under weak aggregation, *i.e.* a small batch size (Zhu et al., 2019).

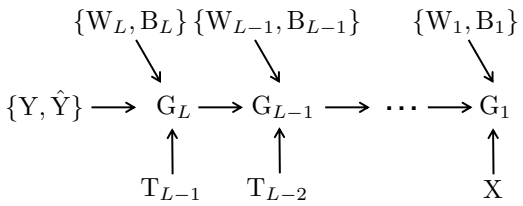

Figure 4: Markov chain from layer L to layer 1 when producing gradients (G) based on weights (W), biases (B), and intermediate representations (T) in backward propagation (Note: $\hat{\mathrm{Y}}$ is T_L, and X is T_0).

A.4 EMPIRICAL SETUP

Here, we explain our experimental evaluation set-up that has been used for the empirical layer-wise characterization and validation of the predictive ability of the proposed metrics.

Goals and validity. Our first evaluation goal is to apply the proposed metrics, $\mathcal{V}$-information and sensitivity, to characterize latent information privacy on DNNs. Our second goal is to establish the validity of the privacy measurements. We leverage the fact that the adversary's success can be an indicator of privacy risks and commonly used in previous research (Yeom et al., 2018; Jayaraman & Evans, 2019; Wei et al., 2020; Mo et al., 2020); intuitively, if an adversary has access to information of higher risk this should yield better attack results. Therefore, for each of the layers of a model, we compare both $\mathcal{V}$-information and sensitivity with the *PIA adversary's AUC score* (*i.e.* area under the ROC curve) when the adversaries have only access to that layer. AUC score is the most common evaluation method for PIAs' success (Powers, 201; Melis et al., 2019). In addition, parameter settings of metrics such as the *chosen attack model* in $\mathcal{V}$-information and *norms* in sensitivity analysis may influence the evaluation results, so they are also measured together in our validations.

Models. For the characterization and validation of the layer-wise privacy risk, we first use two DNN architectures used in (Melis et al., 2019): variational AlexNet and VGG11Net. Both consist of several convolutional (Conv) layers as the earlier layer followed by several fully connected (FC) layers as the latter layers. This type of architecture is widely used because of its good performance; Conv layers capture features of the input data from low to high levels and then following FC layers process classification on them. Specifically, our AlexNet has three Conv layers with 16, 32, 64 filters of 3×3 size (and each has a max-pooling with a size of $(2, 2)$ after), followed by two FC layers with 256 and d_y neurons (where d_y is the output size or the number of classes). VGG11Net has eight Conv layers with 16, 16, 16, 16, 32, 32, 64, 64 filters of 3×3 size (each of the last three has a max-pooling with a size of $(2, 2)$ after), followed by three FC layers with 256, 128, d_y neurons, respectively. In addition, we include another fully connected network (FCNet) which has nine FC layers. All these FC layers have 32 neurons except the output (last) layer. The FCNet architecture has been used in previous research (Shwartz-Ziv & Tishby, 2017; Saxe et al., 2019); by evaluating FCNet we aim to investigate the privacy risk when only FC layers are presented and when layers have the same size but different locations among the model. All three DNNs use ReLU activation functions for all layers (except the output layer).

Datasets. The three models described above are trained on three datasets with attributes, including Labeled Faces in the Wild (LFW) (Huang et al., 2008), Large-scale CelebFaces Attributes (CelebA) (Liu et al., 2015), and Public Figures Face Database (PubFig) (Kumar et al., 2009). LFW contains 13233 face images cropped and resized to 62×47 RGB. All images are labeled with attributes such as gender, race, age, hair color, and more. CelebA contains more than 200k face images of celebrities with 40 attribute annotations such as gender, hair color, eyeglasses, and more. We use a subset of the cropped version (*i.e.* 15000 images) and resize images to 64×64 RGB. We also use a cropped version (100×100 RGB) of PubFig which contains 8300 facial images made up of 100 images for each of 83 persons (Pinto et al., 2011). All images are marked with 73 attributes, *e.g.* gender and race.

Simulation. We conduct our experiments on a cluster with multiple nodes where each has 4 Intel(R) Xeon(R) E5-2620 CPUs (2.00GHz), an NVIDIA RTX 6000 GPU (24GB), and 24GB DDR4 RAM. Deep learning framework Pytorch (Paszke et al., 2019) v1.4.0 is used for privacy measurements, and Theano (Bergstra et al., 2010) v1.0 is used for PIAs. We follow the FL setting in the seminal PIA paper (Melis et al., 2019) for comparing the privacy measurements with PIA results. Specifically, we conduct the learning process using FedSGD as the optimization algorithm. The learning rate is set as 0.01 without momentum. The batch size is 32. The training datasets above are partitioned into two parts to simulate two clients, where one client is assumed as the adversary. As mentioned in (Melis et al., 2019), similar results can be achieved for more clients. The adversary first participates in the training with others for several communication rounds (*e.g.* 100 rounds) and saves snapshots of the received global model to conduct PIAs later. For measuring the ultimate privacy cost, we compute $\mathcal{V}$-information and sensitivity at the end of total communication rounds.